NAME: Jaime Lima

ML 6143 Prof Sundeep Ragan

PRJOECT: NITS CBRS RADAR 3.5 GHz Waveform Detection

RadarWaveformsML - Detecting the Presence of the Waveform Immersed in Gaussian Noise using PCA

Manipulation of Radar and Communication Waveforms using new ML algorithms but not forgetting the Old Tricks

This repository was originally created for project for the Class "Introduction to Machine Learning at NYU Tandon" ministered by Professor Sundeep Ragan (more details at https://github.com/sdrangan/introml (https://github.com/sdrangan/introml))

The main goal is to use synthetic radar waveforms (**RF Dataset of Incumbent Radar Systems in the 3.5 GHz CBRS Band**) provided by the scientists at the US National Institute of Standards and Technology (NIST) (more details at https://data.nist.gov/od/id/mds2-2116 (https://data.nist.gov/od/id/mds2-2116)) to investigate the detection of features in the waveforms using *Machine Learning* techniques.

We are interested in simple trainable algorithms that able to detect the waveforms in the presence of noise (and also fading in the future). We are also interested in find the *signatures* of the waveforms to be able to classify the different radar waveforms and differentiate them from other electromagnetic signals, like the ones used for microwave wireless communications.

Real-time time identification of waveforms is an important tool that can be used to allow coexistence of different emitters sharing the same geographical space. It is also relevant for cyber-security, helping to assure security in cyber-physical environments.

NOTE: to develop the code here we used examples provided in the NIST directory about how read the data files and label files fields correctly in the Python Programming Language. The MATLAB code used by the scientists at NIST to generated radar wavforms simililar to the ones they made avaiable can be found at

https://github.com/usnistgov/SimulatedRadarWaveformGenerator.git/trunk (https://github.com/usnistgov/SimulatedRadarWaveformGenerator.git/trunk)

NIST Data Website https://data.nist.gov/od/id/mds2-2116 (https://data.nist.gov/od/id/mds2-2116)

Contact: Raied Caromi.. Raied Caromi Email: raied.caromi@nist.gov

Importing some Libraries

In [1]:
```
import numpy as np
import h5py
from   pathlib import Path
import pandas as pd
import matplotlib.pyplot as plt
GDRIVE_MOUNTED=False
```

Some Details about How to Dowload RF Dataset of Incumbent Radar Systems in the 3.5 GHz CBRS Band

Each file has 200 synthetically created waveforms, each being 80 ms long. Some waveorms have only gaussian noise. Others have radar waveforms immersered in gaussian noise under several different SNR. Radar waveforms can be different types of simple pulse trains or more complex chirped waveforms.

Each file size is 2.5 GB. The total dataset is 500 GB! We will use just a few files from the database. Additional small csv files provide different types of labels related to the data.

The data can be download directly from NIST using the link https://data.nist.gov/od/id/mds2-2116 (https://data.nist.gov/od/id/mds2-2116)

For convencience, a subset of of the data was downloaded in the Google drive and is accessible for people from NYU. The shared link below provides access to the NYU folks. The shared folder has the same structure as the NIST database but only few of the large waveform files.

https://drive.google.com/drive/folders/1IhVG7mAMoTFjsXbArvrz3cEIXDIJVnDS?usp=sharing (https://drive.google.com/drive/folders/1IhVG7mAMoTFjsXbArvrz3cEIXDIJVnDS?usp=sharing)

Once the code is copied into the person google drive, the code below can be easily adjusted to access the data by chaning the variables *data_folder* and *remote_folder_gdrive*. The code below also allows to run the iphython notebook locally in the machine. In this case the variable *remote_folder_local* needs to be udpated

The code below also checks if GPU is enabled.

In [2]:

```
str_GPU='GPU'
#str_GPU='CPU' #To fake having a GPU
data_folder="SimulatedRadarWaveforms"
remote_folder_gdrive="./gdrive/My Drive/_BIG1/NIST/CBRS/"+data_folder
remote_folder_local="/G/_BIG1/NIST/CBRS/"+data_folder

import os
#!df -k | egrep "Filesystem|overlay"
#!ls

# MOUNT GDRIVE
if 'google.colab' in str(get_ipython()):
  GCOLAB_ENV=True
  print('***RUNNING on CoLab')
  if False==GDRIVE_MOUNTED:
    from google.colab import drive
    drive.mount("/content/gdrive")
    GDRIVE_MOUNTED=True
else:
  print('***NOT RUNNING on CoLab')
  GCOLAB_ENV=False

# CHECK if System has GPU
from tensorflow.python.client import device_lib
resp = device_lib.list_local_devices()
if str_GPU in str(resp):
  has_gpu=True
  gpu_info = !nvidia-smi
  gpu_info = '\n'.join(gpu_info)
  if gpu_info.find('failed') >= 0: print('GPU Info NOT AVAILABLE (!?!)')
  else:                            print(gpu_info)
else:
  has_gpu=False

# LOAD Data
if True==GCOLAB_ENV:
  data_folder_full = remote_folder_gdrive
  #!rsync -av "$remote_folder"   .
  #!du        "$data_folder"
  #!rsync -av "$data_folder"     .
else:
  data_folder_full = remote_folder_local

print('Data Folder:', data_folder_full)
!ls -l     "$data_folder_full"

!rm -rf ./flickr2
if (True==has_gpu):
  print("*** DO SOMETHING (HAS GPU)")
else:
  print("*** DO SOMETHING (HAS NO GPU)")
```

```
***NOT RUNNING on CoLab
Data Folder: /G/_BIG1/NIST/CBRS/SimulatedRadarWaveforms
total 3789
-rwxrwxrwx 1 root root 3864830 Nov 19 08:05 allWaveformsTableCombined.csv
-rwxrwxrwx 1 root root      64 Nov 19 08:05 allWaveformsTableCombined.csv.sha
256
-rwxrwxrwx 1 root root     136 Dec  9 21:40 desktop.ini
dr-xr-xr-x 1 root root    4096 Dec  9 21:40 Group1
dr-xr-xr-x 1 root root       0 Dec  9 21:40 Group2
dr-xr-xr-x 1 root root       0 Dec  9 21:40 Group3
dr-xr-xr-x 1 root root       0 Dec  9 21:40 Group4
-rwxrwxrwx 1 root root    2282 Nov 19 08:04 License.txt
-rwxrwxrwx 1 root root      64 Nov 19 08:04 License.txt.sha256
-rwxrwxrwx 1 root root    1399 Nov 19 08:05 ReadMe.txt
-rwxrwxrwx 1 root root      64 Nov 19 08:04 ReadMe.txt.sha256
dr-xr-xr-x 1 root root    4096 Dec  9 21:40 readWaveformsCodeExamples
*** DO SOMETHING (HAS NO GPU)
```

Reading the Radar Waveform Data Set .mat files and the .csv file with the Additional Labels

This will take some time because the files are large.

In [3]:

```
#dataRootFolder=data_folder
group_No =1  # 1 to 4
subset_No=2 # 1 to 50
# Preps
dataRootFolder  =data_folder_full
fileName        ='group'+str(group_No)+'_subset_'+str(subset_No)+'.mat'
filePath        =Path(dataRootFolder)/('Group'+str(group_No))/(fileName)
waveform_var    ='group'+str(group_No)+'_waveformSubset_'+str(subset_No)
status_var      ='group'+str(group_No)+'_radarStatusSubset_'+str(subset_No)
table_var       ='group'+str(group_No)+'_waveformTableSubset_'+str(subset_No)
infoFileNamePath=Path(dataRootFolder)/('Group'+str(group_No))/('group'+
                      str(group_No)+'_subset_CSVInfo')/(table_var+'.csv')
#
# Read data from .mat file
h5pyObj           = h5py.File(filePath,'r')
subsetSignals     = h5pyObj[waveform_var][()].view(np.complex)
subsetRadarStatus = h5pyObj[status_var  ][()]
#
# Now use Panda to read the info csv file
subsetInfo = pd.read_csv(infoFileNamePath)

print('***DONE')
```

```
***DONE
```

Now Printing Some Fields to Get Familiar with the Data

Two label columns of spectial interest are *subsetRadarStatus=h5pyObj[status_var]* and *

1. h5pyObj[status_varList]: Indicates if a radar waveform is present
2. subsetInfo['BinNo']: Indicates the type of the waveform
3. subsetInfo['SNR']: Indicates the Signal-to-Noise Ratio when the radar waveform is present.

In [4]:

```
def P2R(radii, angles):
    return radii * np.exp(1j*angles)

def R2P(x):
    return np.abs(x), np.angle(x)
```

In [5]:

```
jj = 1.j
print("fileName    =", fileName)
print("waveform_var=", waveform_var)
print("status_var  =", status_var)
print("table_var   =", table_var)
print("h5pyObj     =", h5pyObj)
print('---')
print('[',waveform_var,'   ] subsetSignals.shape=',subsetSignals.shape)
print('[',status_var  ,'] subsetRadarStatus.shape=',subsetRadarStatus.shape)
print('subsetInfo.shape=' ,subsetInfo.shape)
print('---')
print('[',waveform_var,'   ] subsetSignals.shape=',subsetSignals.shape)
NPT=5
r, ang = R2P(subsetSignals[0, 0:NPT])
#print(subsetSignals[sigIndex, 0:NPT])
#print('r=\n', r.T)
#print('ang=\n', ang.T)
print(r*np.exp(jj*ang))
```

```
fileName    = group1_subset_2.mat
waveform_var= group1_waveformSubset_2
status_var  = group1_radarStatusSubset_2
table_var   = group1_waveformTableSubset_2
h5pyObj     = <HDF5 file "group1_subset_2.mat" (mode r)>
---
[ group1_waveformSubset_2    ] subsetSignals.shape= (200, 800000)
[ group1_radarStatusSubset_2 ] subsetRadarStatus.shape= (200, 1)
subsetInfo.shape= (200, 16)
---
[ group1_waveformSubset_2    ] subsetSignals.shape= (200, 800000)
[1.66089269e-07-2.31654912e-07j 1.35645134e-07+2.35493766e-07j
 1.58416011e-07-1.24614912e-07j 1.21105281e-07+4.92919982e-08j
 4.46692447e-07-3.11198016e-07j]
```

In [6]:

```
#%% show the info and status of the first 10
#print(subsetRadarStatus[0:20])
print("LABEL RADAR Waveform Present:\n\n", subsetRadarStatus[0:20].ravel())
print("\nOTHER LABELS below:\n")
print("...LABEL NAMES (Columns):\n... ...", subsetInfo.columns)
print("\n... ... LABELS:\n",subsetInfo[0:20])
#print(subsetInfo['BinNo'][0:20])
aux = subsetInfo['SNR']; aux = aux[~np.isnan(aux)]; aux = np.unique(aux); prin
t(aux);
print('***DONE QUICKLY')
```

```
LABEL RADAR Waveform Present:

 [0 0 1 1 0 0 0 0 0 0 0 0 1 0 1 1 0 1 0 0]

OTHER LABELS below:

...LABEL NAMES (Columns):
... ... Index(['BinNo', 'PulseWidth', 'PulsesPerSecond', 'PulsesPerBurst',
       'ChirpWidth', 'ChirpDirection', 'SamplingFrequency', 'ActualPulseWidt
h',
       'PhaseCodingType', 'SUID', 'radarStatus', 'radarSignalCenterFreq',
       'radarSignalStartTime', 'SNR', 'NoisePowerdBmPerMHz', 'duration'],
      dtype='object')

... ... LABELS:
     BinNo  PulseWidth  PulsesPerSecond  PulsesPerBurst  ChirpWidth  \
0     NaN         NaN              NaN             NaN         NaN
1     NaN         NaN              NaN             NaN         NaN
2   Q3N#3    0.000085            800.0            20.0  50000000.0
3   P0N#2    0.000026           1690.0            10.0         NaN
4     NaN         NaN              NaN             NaN         NaN
5     NaN         NaN              NaN             NaN         NaN
6     NaN         NaN              NaN             NaN         NaN
7     NaN         NaN              NaN             NaN         NaN
8     NaN         NaN              NaN             NaN         NaN
9     NaN         NaN              NaN             NaN         NaN
10    NaN         NaN              NaN             NaN         NaN
11    NaN         NaN              NaN             NaN         NaN
12  Q3N#1    0.000003           1620.0            20.0  50000000.0
13    NaN         NaN              NaN             NaN         NaN
14  Q3N#1    0.000004            870.0            22.0  70000000.0
15  P0N#1    0.000002           1010.0            35.0         NaN
16    NaN         NaN              NaN             NaN         NaN
17  Q3N#2    0.000021           1750.0             2.0   3000000.0
18    NaN         NaN              NaN             NaN         NaN
19    NaN         NaN              NaN             NaN         NaN

   ChirpDirection  SamplingFrequency  ActualPulseWidth PhaseCodingType  \
0             NaN           10000000               NaN             NaN
1             NaN           10000000               NaN             NaN
2            Down           10000000          0.000085             NaN
3             NaN           10000000          0.000026          Barker
4             NaN           10000000               NaN             NaN
5             NaN           10000000               NaN             NaN
6             NaN           10000000               NaN             NaN
7             NaN           10000000               NaN             NaN
8             NaN           10000000               NaN             NaN
9             NaN           10000000               NaN             NaN
10            NaN           10000000               NaN             NaN
11            NaN           10000000               NaN             NaN
12           Down           10000000          0.000003             NaN
13            NaN           10000000               NaN             NaN
14             Up           10000000          0.000004             NaN
15            NaN           10000000          0.000002             NaN
16            NaN           10000000               NaN             NaN
17           Down           10000000          0.000021             NaN
18            NaN           10000000               NaN             NaN
```

```
19            NaN           10000000               NaN             NaN

                                    SUID  radarStatus  radarSignalCenterFreq  \
0                                    NaN            0                    NaN
1                                    NaN            0                    NaN
2   1c19ef39-f88b-4ff9-a77d-68bc9eb141ef            1                    0.0
3   b6e3b197-b0ff-49d2-916c-5bc60df615e4            1              1623597.0
4                                    NaN            0                    NaN
5                                    NaN            0                    NaN
6                                    NaN            0                    NaN
7                                    NaN            0                    NaN
8                                    NaN            0                    NaN
9                                    NaN            0                    NaN
10                                   NaN            0                    NaN
11                                   NaN            0                    NaN
12  6ac0e889-f818-49de-9e7b-40779815a572            1                    0.0
13                                   NaN            0                    NaN
14  6d0b9cf4-8efd-4e2c-bc77-139f0c4a8054            1                    0.0
15  00c8c6d2-064b-4396-b340-b0ce8d321716            1              -705689.0
16                                   NaN            0                    NaN
17  f9643519-bda0-47e9-9ea0-9745406539c6            1             -1046091.0
18                                   NaN            0                    NaN
19                                   NaN            0                    NaN

    radarSignalStartTime   SNR  NoisePowerdBmPerMHz  duration
0                    NaN   NaN                 -109      0.08
1                    NaN   NaN                 -109      0.08
2               0.008170  10.0                 -109      0.08
3               0.015718  18.0                 -109      0.08
4                    NaN   NaN                 -109      0.08
5                    NaN   NaN                 -109      0.08
6                    NaN   NaN                 -109      0.08
7                    NaN   NaN                 -109      0.08
8                    NaN   NaN                 -109      0.08
9                    NaN   NaN                 -109      0.08
10                   NaN   NaN                 -109      0.08
11                   NaN   NaN                 -109      0.08
12              0.018937  16.0                 -109      0.08
13                   NaN   NaN                 -109      0.08
14              0.020553  14.0                 -109      0.08
15              0.022690  16.0                 -109      0.08
16                   NaN   NaN                 -109      0.08
17              0.022535  16.0                 -109      0.08
18                   NaN   NaN                 -109      0.08
19                   NaN   NaN                 -109      0.08
[10. 12. 14. 16. 18. 20.]
***DONE QUICKLY
```

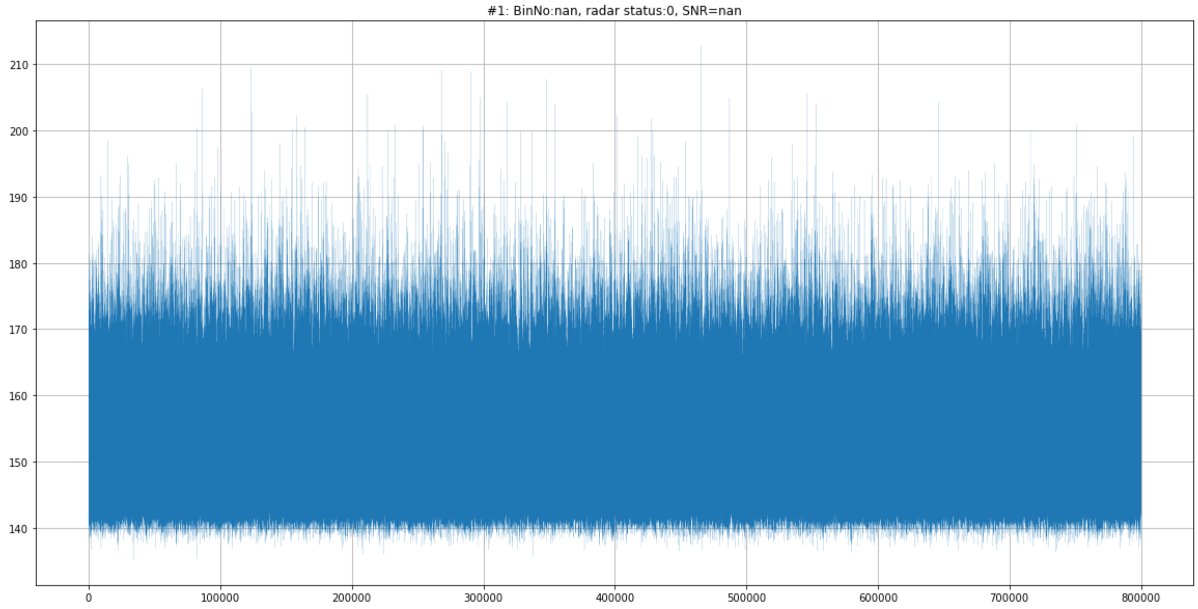
Below we plot three waveforms in the time domain.

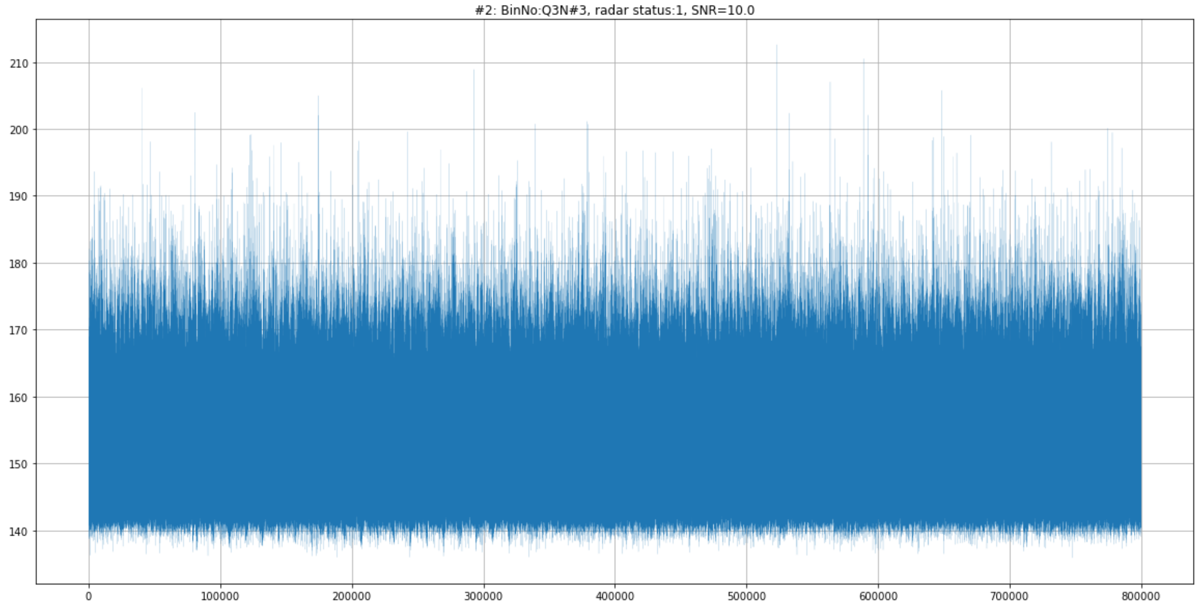
The first has only noise, the second has a relatively low SNR and the thrid a relatively higher SNR. Note that the signal can be detected and it seems that only noise is present.

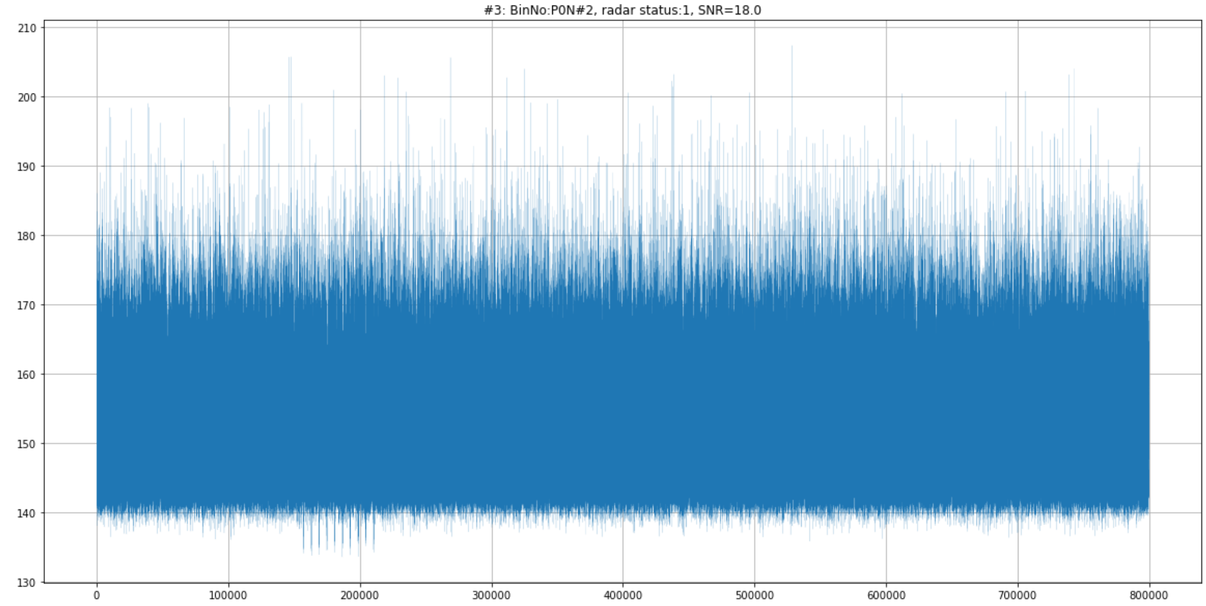
In [7]:

```
noExamples=3
startFrom=1
for sigIndex in range(startFrom,startFrom+noExamples):
      #sigIndex=6 # Matlab index 6
      print('***Doing for sigIndex=', sigIndex)
      plt.figure(figsize=(20,10))
      plt.plot(-10*np.log(np.absolute(subsetSignals[sigIndex][0:-1])),linewidth=0.1)
      plt.title('#'+str(sigIndex)+': BinNo:'+str(subsetInfo['BinNo'][sigIndex])+', radar status:'+str(subsetInfo['radarStatus'][sigIndex])+', SNR='+str(subsetInfo['SNR'][sigIndex]))
      plt.grid()
print('***DONE!')
```

```
***Doing for sigIndex= 1
***Doing for sigIndex= 2
***Doing for sigIndex= 3
***DONE!
```







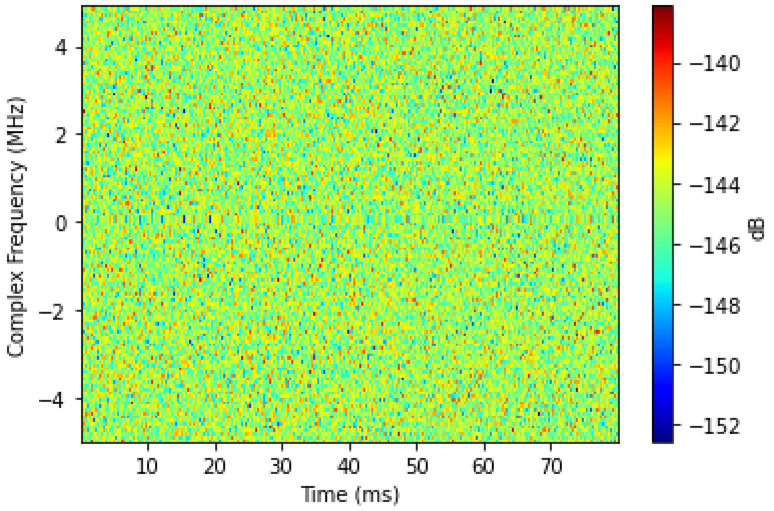
Below we Plot Three Waveforms in the Frequency vs Time Domain.

Again the first has only noise, the second has a relatively low SNR and the thrid a relatively higher SNR. Note that now the signal is resolved cells of time and frequency the signal can be seen as non-random cell patterns on top of the 'sea of noise'. It is hardly noticiable in the second case (SNR=10 dB), but still it can be seen lile a 'cat scratch' o lower left quadrant of the spectogram. In the case of the third spectorgram, the signal is much easier to see because of the higer SNR and also because the specific waveform used is short and 'shines more' above the noise 'sea level'

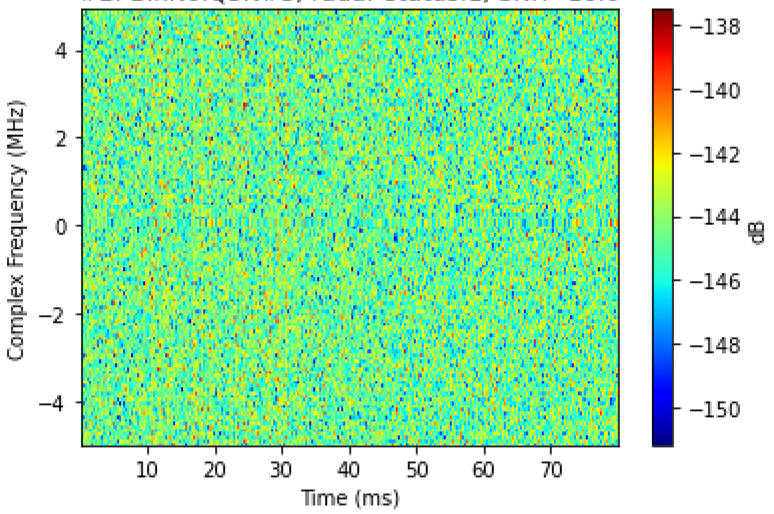
In [11]:

```
from scipy import signal
Nfft=128
groupby=16  # no. of consectuive FFTs over which to take max

noExamples=3
startFrom=1
for sigIndex in range(startFrom,startFrom+noExamples):
  f, t0, S0 = signal.spectrogram(subsetSignals[sigIndex], fs=10e6, nperseg=Nfft, scaling='spectrum', return_onesided=False)
  #
  S0[0:1, :] = S0[1:2, :]
  L = S0.shape[1]/groupby
  S1 = np.reshape(S0[:,:int(L)*groupby], (Nfft,int(L),groupby))
  S = np.amax(S1, axis=-1)
  t = t0[groupby-1::groupby]

  print('S.shape=', S.shape)
  #S[0:1, :] = np.mean(S[70:80,:], axis=0)
  #S[0:1, :] = S[1:2, :]
  fshi = np.fft.fftshift(f/1e6)
  Sshi = np.fft.fftshift(10.*np.log10(S), axes=0)
  #print('Sshi.shape=', Sshi.shape
  #
  #fig = plt.figure(figsize=(10,10))
  fig = plt.figure()
  cax = plt.pcolormesh(t*1e3, fshi, Sshi, cmap='jet')
  plt.title('#'+str(sigIndex)+': BinNo:'+str(subsetInfo['BinNo'][sigIndex])+', radar status:'+str(subsetInfo['radarStatus'][sigIndex])+', SNR='+str(subsetInfo['SNR'][sigIndex]))
  plt.xlabel('Time (ms)')
  plt.ylabel('Complex Frequency (MHz)')
  fig.colorbar(cax).set_label('dB')

print('***DONE!')
```

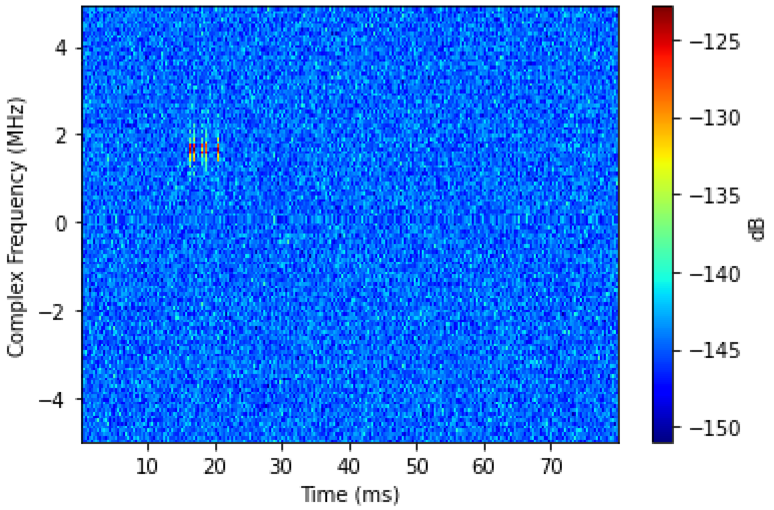
```
S.shape= (128, 446)
S.shape= (128, 446)
S.shape= (128, 446)
***DONE!
```



#1: BinNo:nan, radar status:0, SNR=nan



#2: BinNo:Q3N#3, radar status:1, SNR=10.0



#3: BinNo:P0N#2, radar status:1, SNR=18.0

In [12]:

```
from scipy import signal
nruns=20
pca0 = 1
#noExamples=3
#startFrom=6
Nfft=128
groupby=16  # no. of consectuive FFTs over which to take max
#find signals with high SNR, e.g. SNR=20 dB
#highSNRIndex=np.where(subsetInfo.SNR==20)
#find signals with high SNR, e.g. SNR=10 dB

# [10. 12. 14. 16. 18. 20.]
#IselWaves=np.where(np.isnan(subsetInfo.SNR)) #Noise Only
#IselWaves=np.where(subsetInfo.SNR==10)
#IselWaves=np.where(subsetInfo.SNR==14)
#IselWaves=np.where(subsetInfo.SNR==16)
#IselWaves=np.where(subsetInfo.SNR==18)
IselWaves=np.where(subsetInfo.SNR==20)

idx=1
for I in np.nditer(IselWaves):
  if idx > nruns: break
  sigIndex=int(I)
  print('***-------------------')
  print('***Doing for sigIndex=', sigIndex, 'I=', I)
  print('...NSD   :', subsetInfo['NoisePowerdBmPerMHz'][sigIndex])
  print('...SNR   :', subsetInfo['SNR'][sigIndex])
  print('...BinNo :', subsetInfo['BinNo'][sigIndex])
  #
  f, t0, S0 = signal.spectrogram(subsetSignals[sigIndex], fs=10e6, nperseg=Nff
t, scaling='spectrum', return_onesided=False)
  #
  S0[0:1, :] = S0[1:2, :]
  L = S0.shape[1]/groupby
  S1 = np.reshape(S0[:,:int(L)*groupby], (Nfft,int(L),groupby))
  S = np.amax(S1, axis=-1)
  t = t0[groupby-1::groupby]

  print('S.shape=', S.shape)
  #S[0:1, :] = np.mean(S[70:80,:], axis=0)
  #S[0:1, :] = S[1:2, :]
  fshi = np.fft.fftshift(f/1e6)
  Sshi = np.fft.fftshift(10.*np.log10(S), axes=0)
  #print('Sshi.shape=', Sshi.shape)

  if 0: #Controls the printing of the Spectograms
    fig = plt.figure()
    cax = plt.pcolormesh(Sshi.T.dot(Sshi), cmap='jet')
    plt.title('Run# '+str(idx)+' '+str(subsetInfo['BinNo'][sigIndex])+' TIME')
    plt.xlabel('Time (ms)')
    plt.ylabel('Complex Frequency (MHz)')
    fig.colorbar(cax).set_label('dB')
    #
    fig = plt.figure()
    cax = plt.pcolormesh(Sshi.dot(Sshi.T), cmap='jet')
    plt.title('Run# '+str(idx)+' '+str(subsetInfo['BinNo'][sigIndex])+' TIME')
    plt.xlabel('Time (ms)')
    plt.ylabel('Complex Frequency (MHz)')
    fig.colorbar(cax).set_label('dB')
    #

  if 1: #Controls the printing of the Spectograms
    fig = plt.figure()
    cax = plt.pcolormesh(t*1e3, fshi, Sshi, cmap='jet')
    plt.title('Run# '+str(idx)+' '+str(subsetInfo['BinNo'][sigIndex])+' TIME')
    plt.xlabel('Time (ms)')
    plt.ylabel('Complex Frequency (MHz)')
    fig.colorbar(cax).set_label('dB')
    #

  if 1: #Controls the printing of the stem with the PCAs
    ##u, s, vh = np.linalg.svd(Sshi, full_matrices=False);
    #u, s, vh = np.linalg.svd(Sshi.T, full_matrices=False);
    u, s, vh = np.linalg.svd(Sshi.T.dot(Sshi), full_matrices=False);
    aux = 100*np.sort(s)[::-1]/np.sum(s)
    print(aux[0+pca0:10])
    fig = plt.figure()
    #plt.stem(np.log10(s[0+pca0:50]))
    plt.stem(aux[0+pca0:50])
    plt.title('Run# '+str(idx)+' '+str(subsetInfo['BinNo'][sigIndex])+' FREQ')

  print('...#%02d'%sigIndex,':','Min =', 10.*np.log10(min(S.fla
t)),'\n.........Max =', 10.*np.log10(max(S.flat)))
  idx += 1;
  # end of for()

print('***DONE!')
```

```
***-------------------
***Doing for sigIndex= 35 I= 35
...NSD   : -109
...SNR   : 20.0
...BinNo : P0N#2
S.shape= (128, 446)
[0.00232399 0.00022074 0.00020754 0.00019675 0.00019444 0.00019216
 0.00018671 0.00018247 0.000178  ]
...#35 : Min = -151.02472837414226
.........Max = -120.08953140163626
***-------------------
***Doing for sigIndex= 58 I= 58
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#3
```

```
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])
```

```
S.shape= (128, 446)
[0.00028152 0.00024076 0.00023615 0.00022128 0.00021119 0.00019971
 0.00019233 0.00018658 0.00018577]
```

```
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])
```

```
...#58 : Min = -151.3335135591741
.........Max = -131.94023891783357
***-------------------
***Doing for sigIndex= 65 I= 65
...NSD   : -109
...SNR   : 20.0
...BinNo : P0N#1
S.shape= (128, 446)
[0.00058672 0.00023749 0.00020762 0.00019967 0.00019248 0.00018561
 0.00018214 0.00018023 0.00017686]
...#65 : Min = -151.03655666254565
.........Max = -135.06969963599408
***-------------------
***Doing for sigIndex= 68 I= 68
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#1
S.shape= (128, 446)
```

```
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])
```

```
[0.00149946 0.00023857 0.00022552 0.00021438 0.00020129 0.00019603
 0.00019054 0.00018435 0.00018042]
...#68 : Min = -151.48232264681317
.........Max = -131.4430502721453
***-------------------
***Doing for sigIndex= 91 I= 91
...NSD   : -109
...SNR   : 20.0
...BinNo : P0N#2
```

```
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])
```

```
S.shape= (128, 446)
[0.0014196  0.00020548 0.00020319 0.00019533 0.00019233 0.00018925
 0.00018322 0.0001826  0.00017847]
...#91 : Min = -150.86594370624715
.........Max = -122.10499582954179
***-------------------
***Doing for sigIndex= 111 I= 111
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#2
```

```
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])
```

```
S.shape= (128, 446)
[0.00028613 0.00025746 0.00022483 0.00020191 0.00019263 0.00018966
 0.00018868 0.00018645 0.00018143]
...#111 : Min = -151.81896662484968
.........Max = -134.30213818061887
***-------------------
***Doing for sigIndex= 114 I= 114
...NSD   : -109
...SNR   : 20.0
...BinNo : P0N#1
```

```
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])

S.shape= (128, 446)
[0.00097962 0.0002291  0.00020788 0.00020138 0.00019358 0.00018798
 0.00018559 0.00017924 0.00017562]

<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])

...#114 : Min = -151.19106127468288
.........Max = -133.5301606458608
***-------------------
***Doing for sigIndex= 115 I= 115
...NSD   : -109
...SNR   : 20.0
...BinNo : P0N#1
S.shape= (128, 446)
[0.00114792 0.00020157 0.00019883 0.00019626 0.00018989 0.00018381
 0.00018349 0.00017746 0.00017502]

<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])

...#115 : Min = -151.44744722793197
.........Max = -132.7352663963266
***-------------------
***Doing for sigIndex= 117 I= 117
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#2
S.shape= (128, 446)
[0.00157415 0.00024911 0.00022108 0.0002062  0.0002032  0.00018783
 0.00018305 0.00018187 0.0001738 ]

<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])
```

```
...#117 : Min = -151.07777991582591
.........Max = -125.53220642980008
***-------------------
***Doing for sigIndex= 119 I= 119
...NSD   : -109
...SNR   : 20.0
...BinNo : P0N#1
S.shape= (128, 446)
[0.00059376 0.00022143 0.00021401 0.00020229 0.00019418 0.00018841
 0.00018659 0.00018389 0.00018114]

<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])

...#119 : Min = -151.26911147179163
.........Max = -135.12135248267165
***-------------------
***Doing for sigIndex= 124 I= 124
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#2
S.shape= (128, 446)

<ipython-input-12-e930dbe3b524>:62: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:76: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])

[0.0004879  0.0002956  0.00023166 0.00020239 0.00019962 0.00019502
 0.00019155 0.00018967 0.00018455]
...#124 : Min = -150.87422512197674
.........Max = -131.52685010457566
***-------------------
***Doing for sigIndex= 125 I= 125
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#1
S.shape= (128, 446)
```

```
<ipython-input-12-e930dbe3b524>:62: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:76: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])
```

```
[0.00372633 0.0002212  0.00020698 0.00020187 0.00019325 0.00018656
 0.00018463 0.00017852 0.00017655]
...#125 : Min = -151.38356171593816
.........Max = -127.53079386440504
***-------------------
***Doing for sigIndex= 139 I= 139
...NSD   : -109
...SNR   : 20.0
...BinNo : P0N#1
S.shape= (128, 446)
```

```
<ipython-input-12-e930dbe3b524>:62: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:76: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])
```

```
[0.00094953 0.00021976 0.00020538 0.00020013 0.00019493 0.00018718
 0.00018442 0.00018219 0.00017843]
...#139 : Min = -150.7372754920851
.........Max = -133.06277251375909
***-------------------
***Doing for sigIndex= 141 I= 141
...NSD   : -109
...SNR   : 20.0
...BinNo : P0N#2
S.shape= (128, 446)
```

```
<ipython-input-12-e930dbe3b524>:62: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:76: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])

[0.00147385 0.00023513 0.0002092  0.00019916 0.00019195 0.00018956
 0.00018602 0.00018301 0.0001774 ]
...#141 : Min = -151.3084014549656
.........Max = -121.11324542589622
***-------------------
***Doing for sigIndex= 147 I= 147
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#3
S.shape= (128, 446)

<ipython-input-12-e930dbe3b524>:62: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:76: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])

[0.00023904 0.0002188  0.00020509 0.00019901 0.00019633 0.00019169
 0.00018638 0.00018121 0.00017851]
...#147 : Min = -151.71647924571732
.........Max = -131.90681479594087
***-------------------
***Doing for sigIndex= 167 I= 167
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#1
S.shape= (128, 446)
```

```
<ipython-input-12-e930dbe3b524>:62: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()

[0.00103332 0.00023766 0.00023341 0.00019999 0.00019372 0.00019179
 0.00018909 0.00018516 0.00018325]
...#167 : Min = -151.25604488973826
.........Max = -131.0912792080857
***-------------------
***Doing for sigIndex= 178 I= 178
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#3

<ipython-input-12-e930dbe3b524>:76: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])

S.shape= (128, 446)
[0.00023161 0.00019792 0.0001898  0.00018805 0.00018505 0.00018181
 0.0001789  0.00017713 0.00017481]

<ipython-input-12-e930dbe3b524>:62: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:76: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])

...#178 : Min = -151.38530436436767
.........Max = -134.32642947361796
***-------------------
***Doing for sigIndex= 183 I= 183
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#2
S.shape= (128, 446)
```

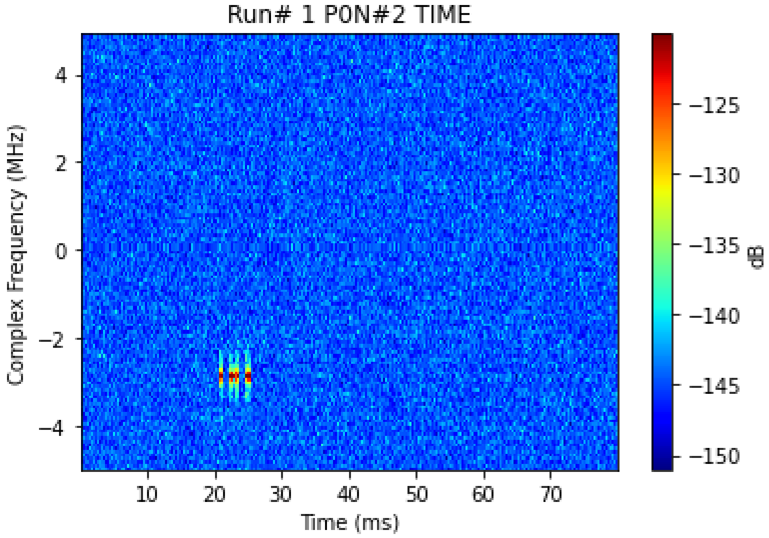
```
<ipython-input-12-e930dbe3b524>:62: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:76: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])
```

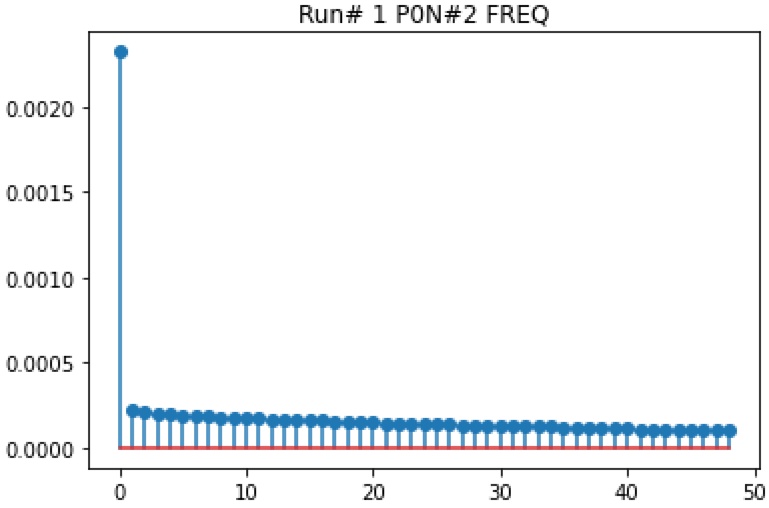
```
[0.00030974 0.00024871 0.00021824 0.00020223 0.00019935 0.00019302
 0.00018828 0.00018378 0.00017726]
...#183 : Min = -151.24298325530708
.........Max = -133.47340456090927
***-------------------
***Doing for sigIndex= 184 I= 184
...NSD   : -109
...SNR   : 20.0
...BinNo : P0N#2
S.shape= (128, 446)
```

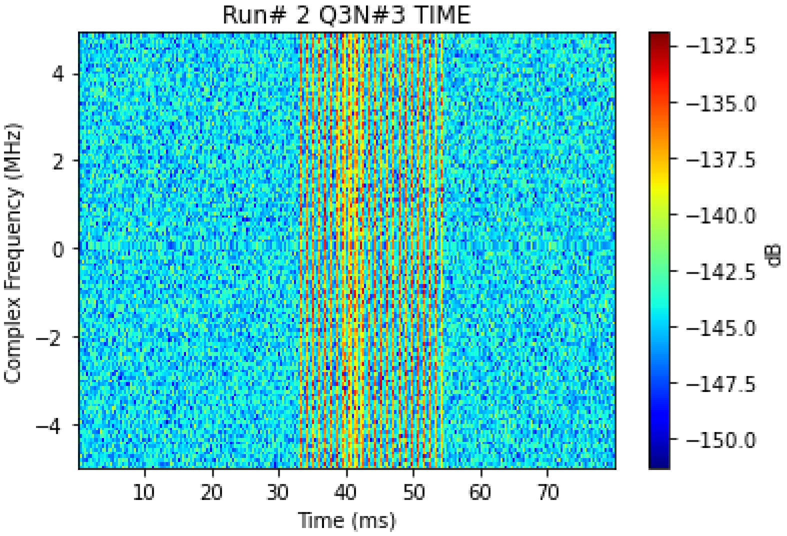
```
<ipython-input-12-e930dbe3b524>:62: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:76: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])
```

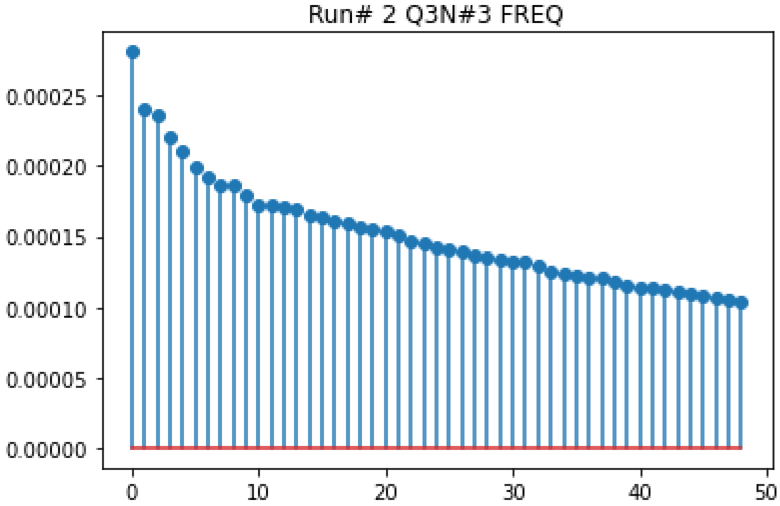
```
[0.001358   0.00021907 0.00020433 0.00019646 0.0001938  0.00019228
 0.00018693 0.00018027 0.00017898]
...#184 : Min = -151.502286502086
.........Max = -124.96510235620038
***-------------------
***Doing for sigIndex= 196 I= 196
...NSD   : -109
...SNR   : 20.0
...BinNo : Q3N#2
S.shape= (128, 446)
[0.000642   0.00024194 0.0001981  0.00019632 0.00018952 0.00018555
 0.00018321 0.00017891 0.00017742]
```

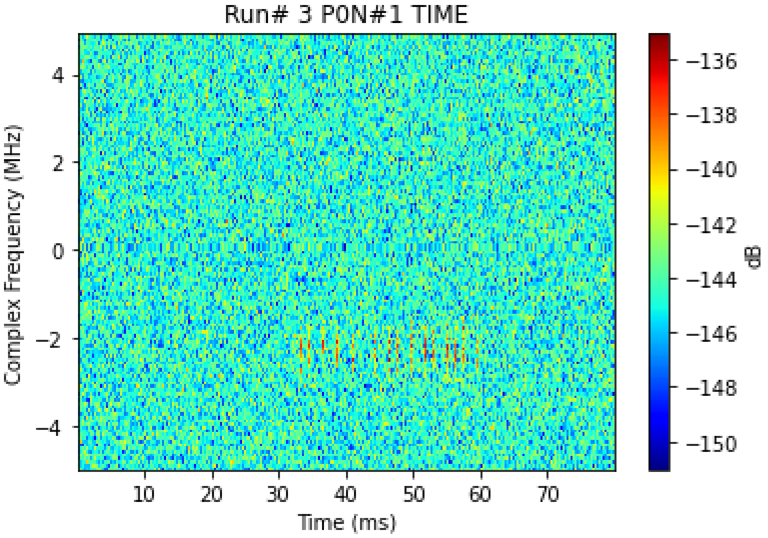
```
<ipython-input-12-e930dbe3b524>:62: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:76: RuntimeWarning: More than 20 figures have
been opened. Figures created through the pyplot interface (`matplotlib.pyplo
t.figure`) are retained until explicitly closed and may consume too much memo
ry. (To control this warning, see the rcParam `figure.max_open_warning`).
  fig = plt.figure()
<ipython-input-12-e930dbe3b524>:78: UserWarning: In Matplotlib 3.3 individual
lines on a stem plot will be added as a LineCollection instead of individual
lines. This significantly improves the performance of a stem plot. To remove
this warning and switch to the new behaviour, set the "use_line_collection" k
eyword argument to True.
  plt.stem(aux[0+pca0:50])

...#196 : Min = -151.40688066030864
.........Max = -125.26483775092797
***DONE!
```

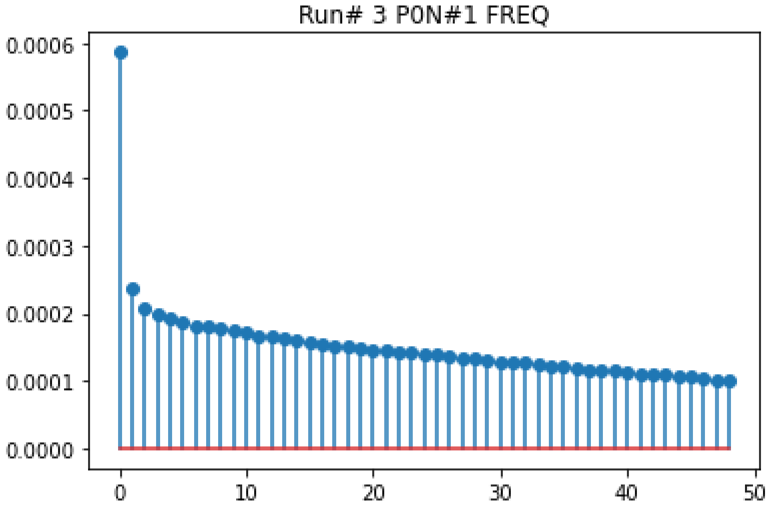


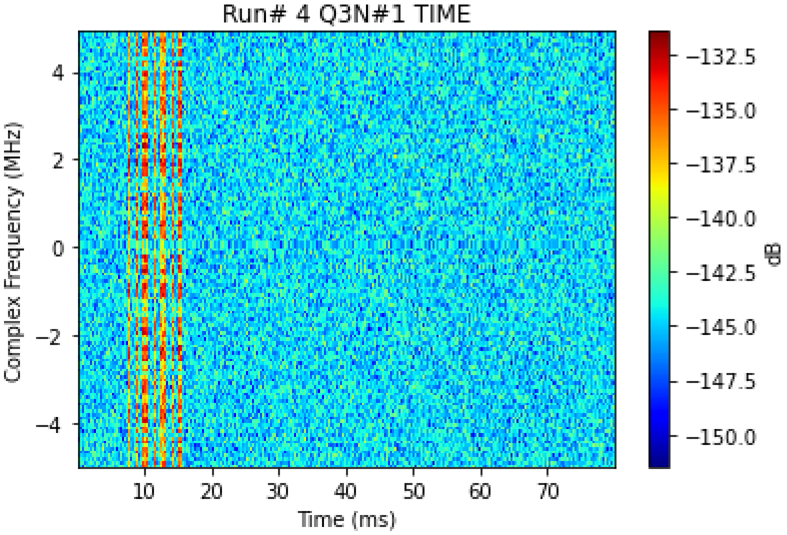


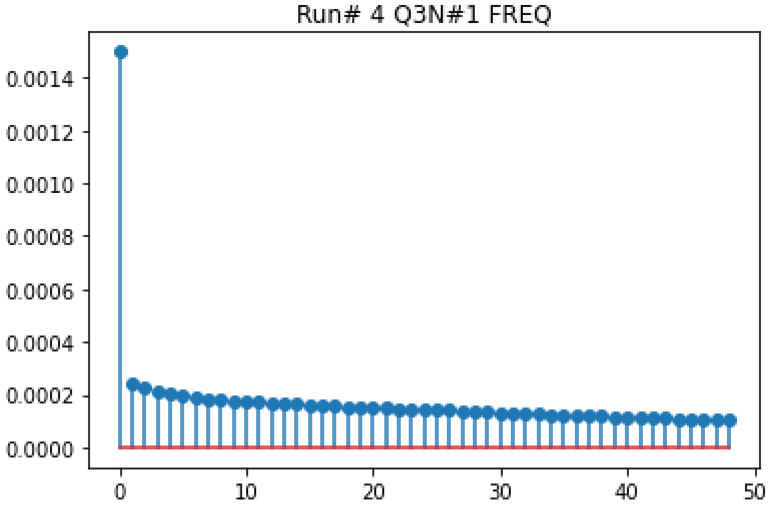

Run# 2 Q3N#3 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−132.5
−135.0
−137.5
−140.0
−142.5
−145.0
−147.5
−150.0
dB

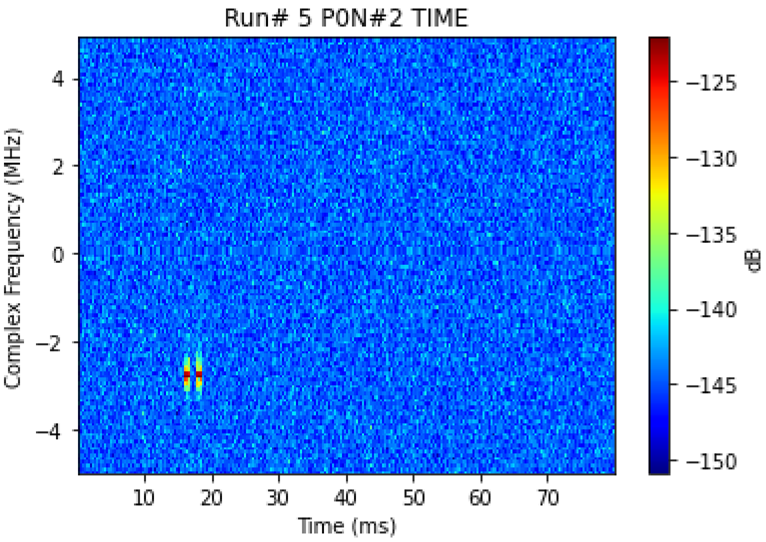

Run# 2 Q3N#3 FREQ
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
0
10
20
30
40
50

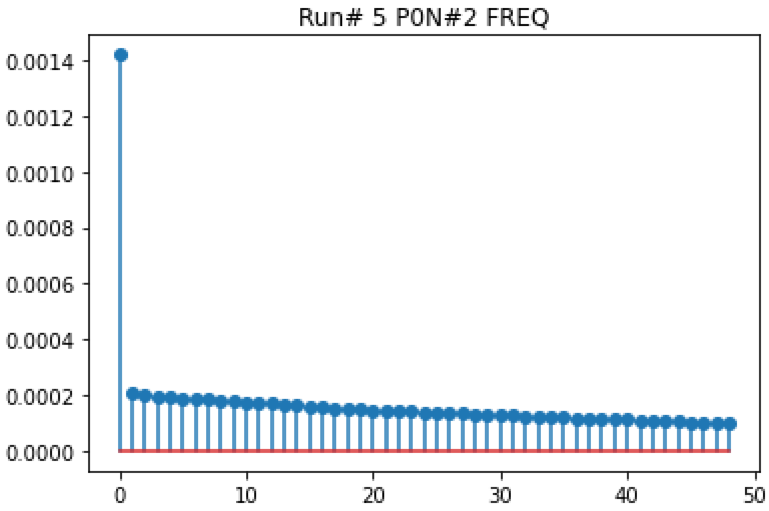

Run# 3 P0N#1 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−136
−138
−140
−142
−144
−146
−148
−150
dB

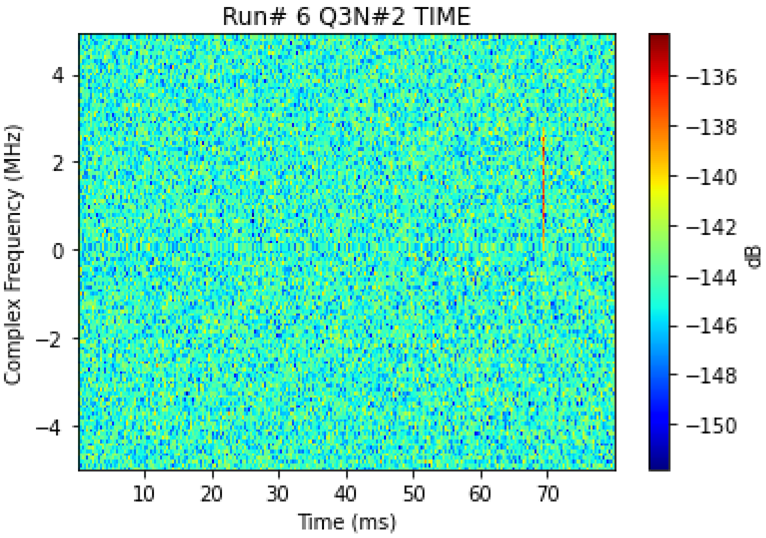

Run# 3 P0N#1 FREQ
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
10
20
30
40
50

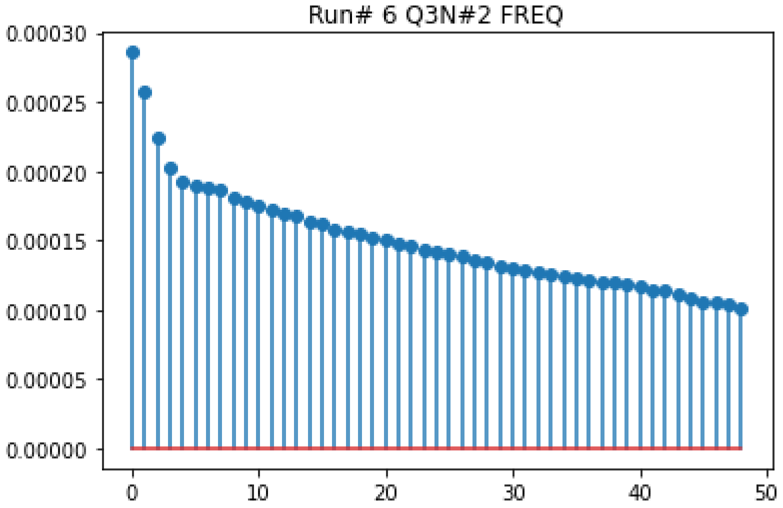

Run# 4 Q3N#1 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−132.5
−135.0
−137.5
−140.0
−142.5
−145.0
−147.5
−150.0
dB

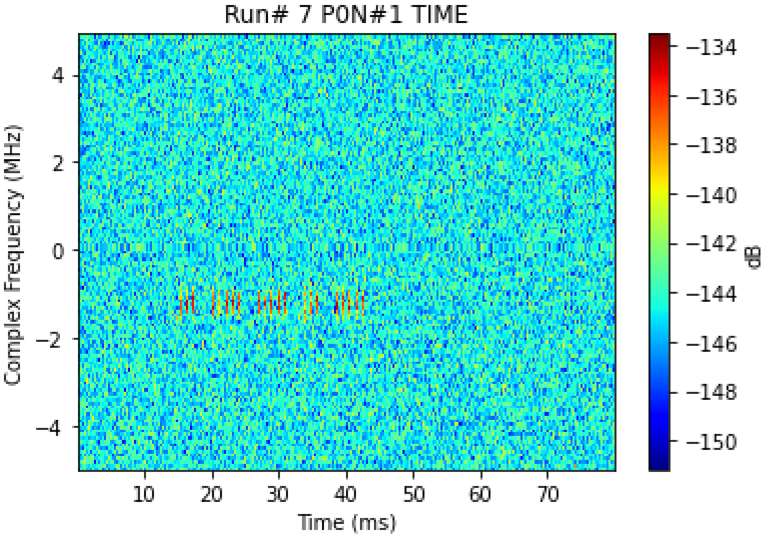

Run# 4 Q3N#1 FREQ
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
10
20
30
40
50

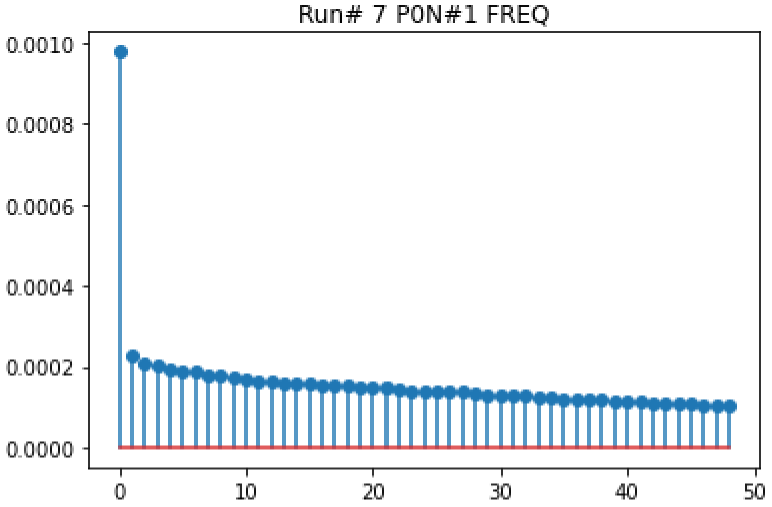

Run# 5 P0N#2 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−125
−130
−135
−140
−145
−150
dB

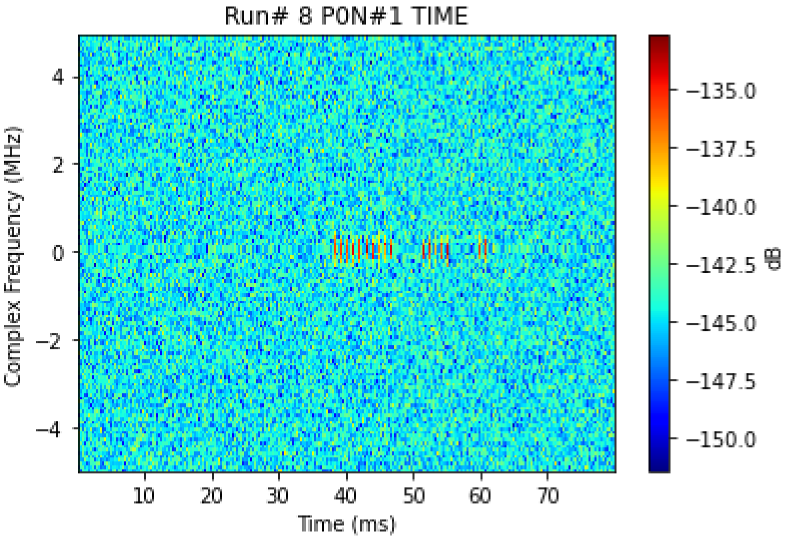

Run# 5 P0N#2 FREQ
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
10
20
30
40
50

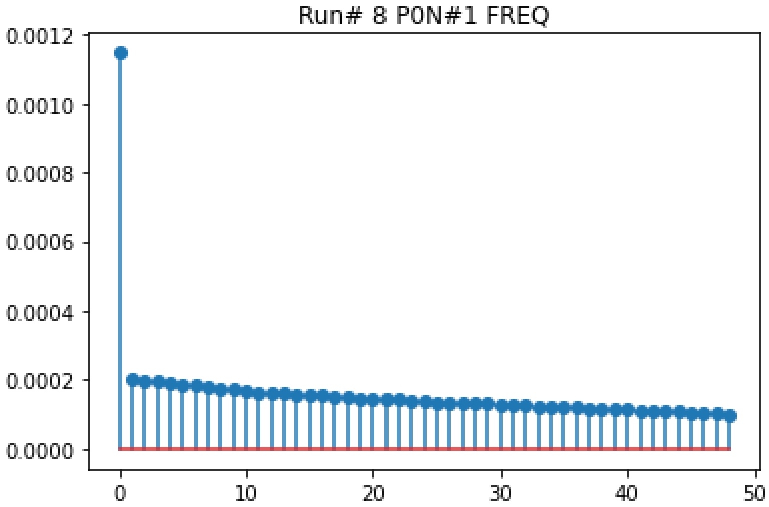

Run# 6 Q3N#2 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−136
−138
−140
−142
−144
−146
−148
−150
dB

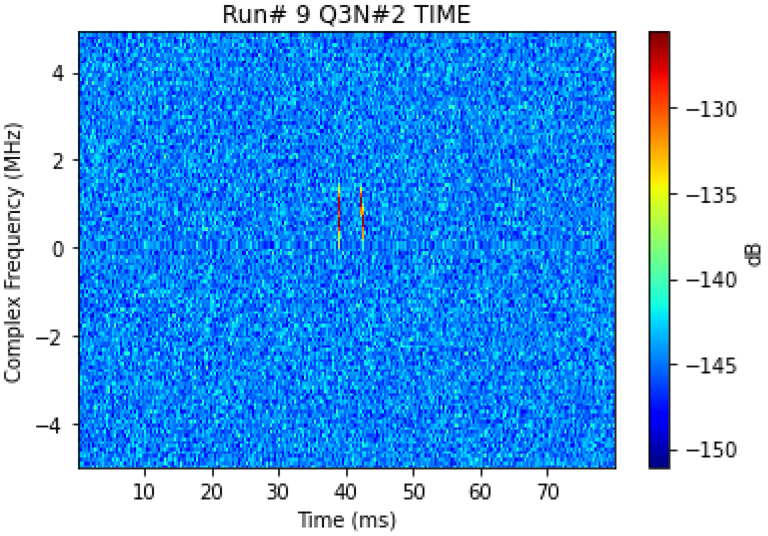

Run# 6 Q3N#2 FREQ
0.00030
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
0
10
20
30
40
50

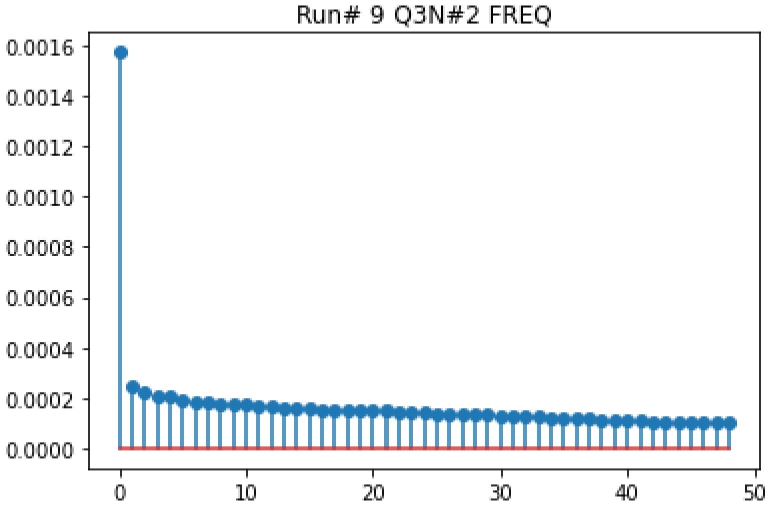

Run# 7 P0N#1 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−134
−136
−138
−140
−142
−144
−146
−148
−150
dB

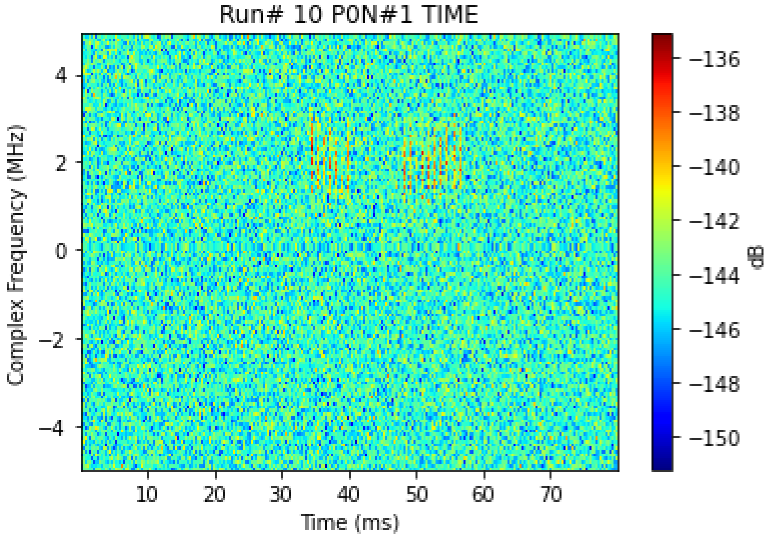

Run# 7 P0N#1 FREQ
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
10
20
30
40
50

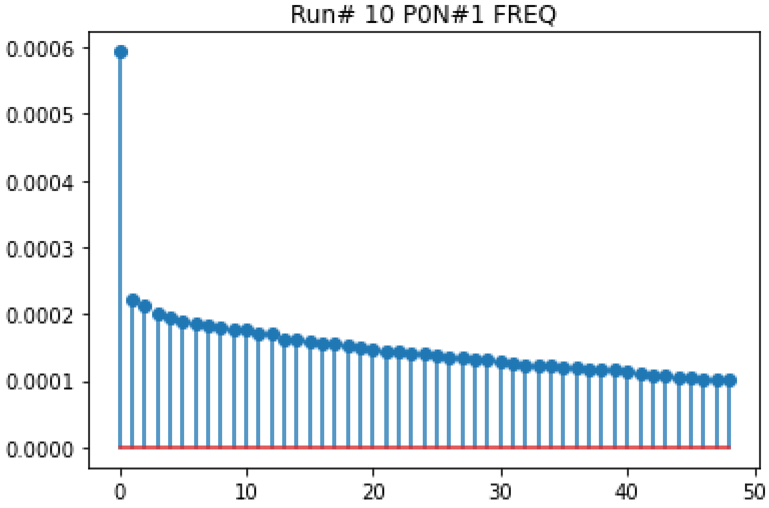

Run# 8 P0N#1 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−135.0
−137.5
−140.0
−142.5
−145.0
−147.5
−150.0
dB

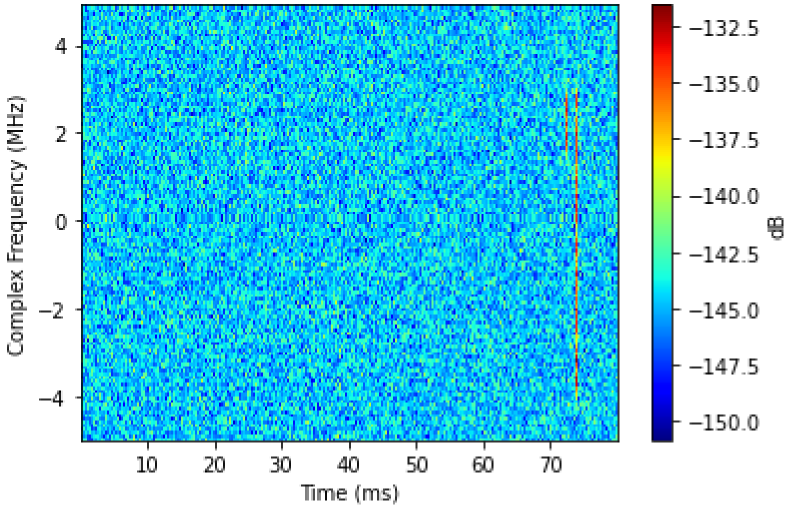

Run# 8 P0N#1 FREQ
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
10
20
30
40
50

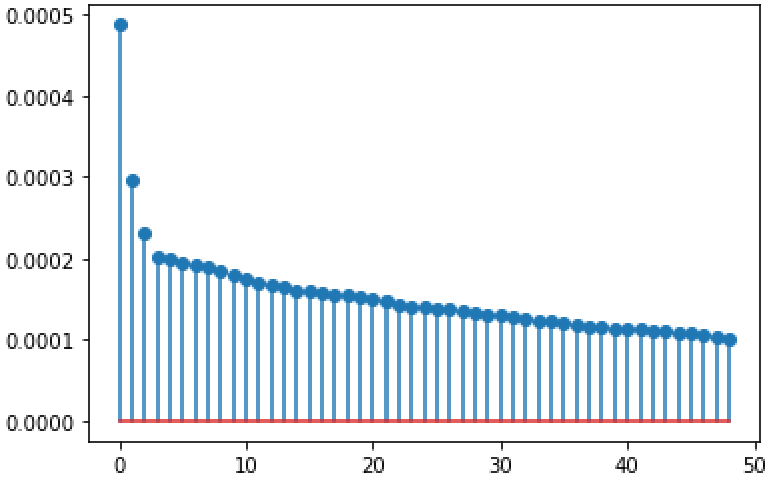

Run# 9 Q3N#2 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−130
−135
−140
−145
−150
dB

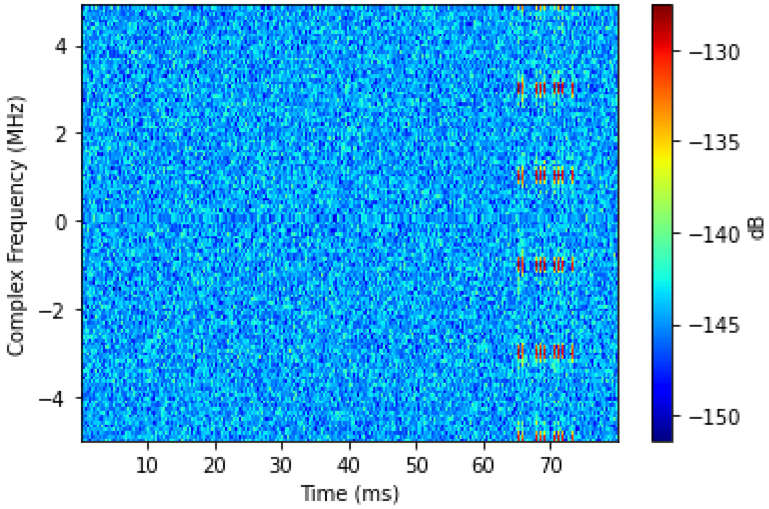

Run# 9 Q3N#2 FREQ
0.0016
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
10
20
30
40
50

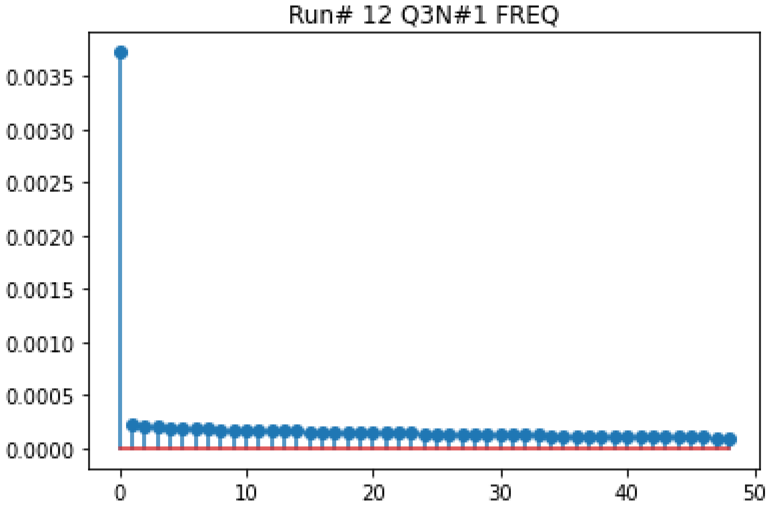

Run# 10 P0N#1 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−136
−138
−140
−142
−144
−146
−148
−150
dB

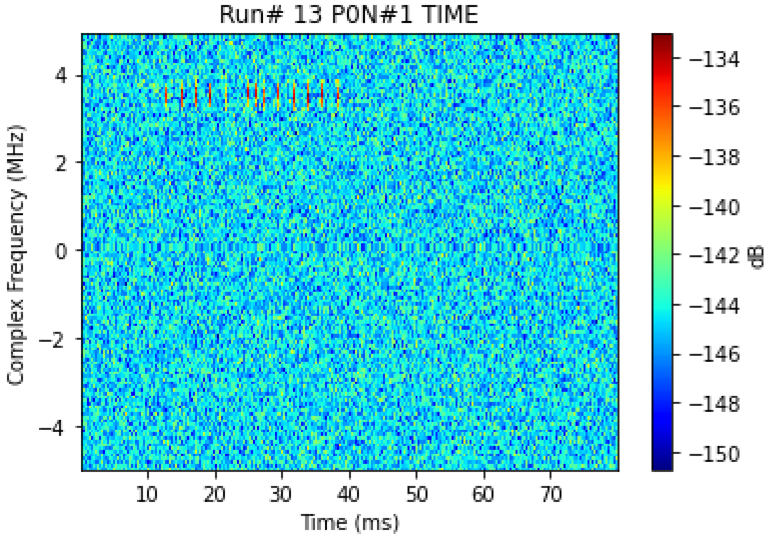

Run# 10 P0N#1 FREQ
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
10
20
30
40
50

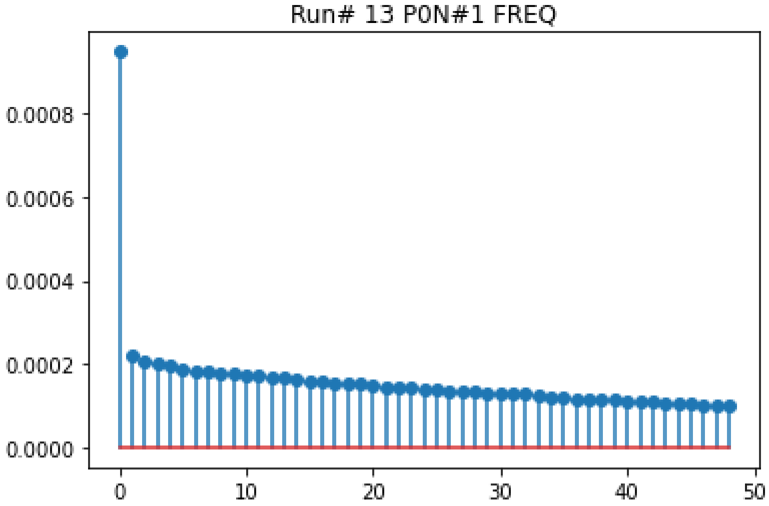

Run# 11 Q3N#2 TIME

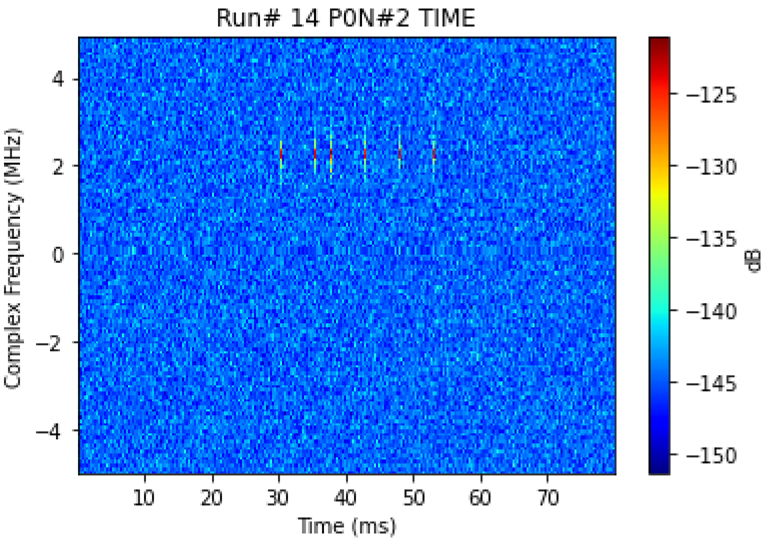

Run# 11 Q3N#2 FREQ

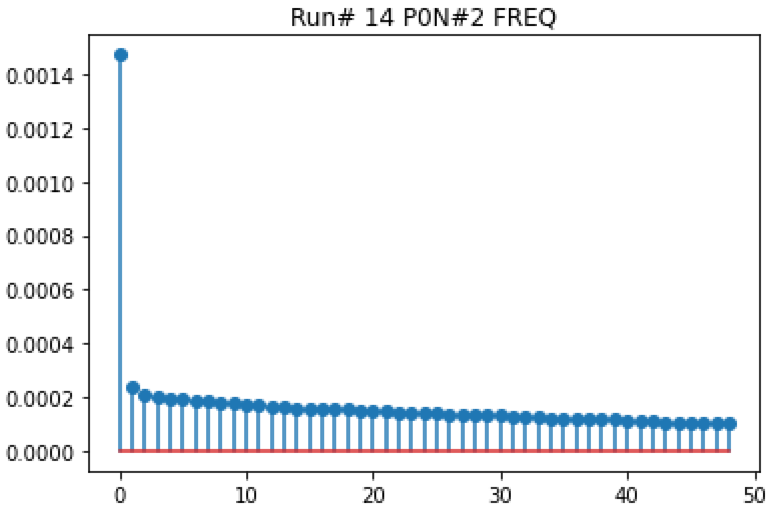

Run# 12 Q3N#1 TIME

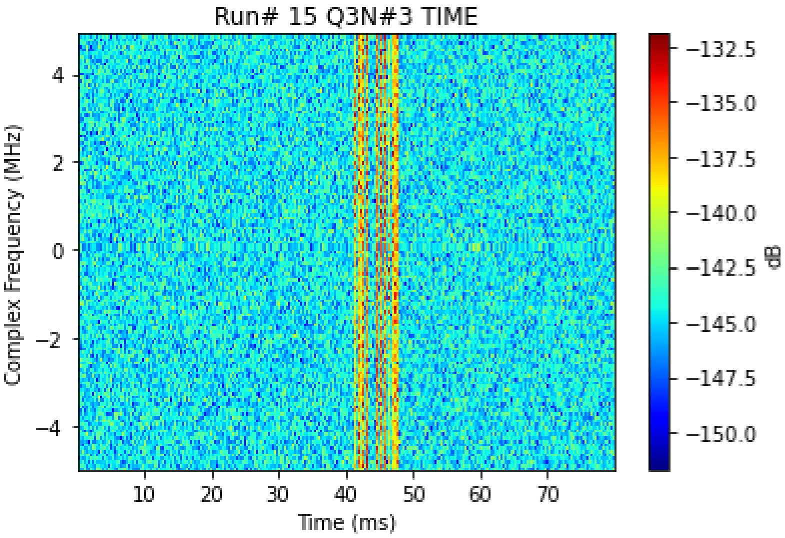

Run# 12 Q3N#1 FREQ
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
10
20
30
40
50

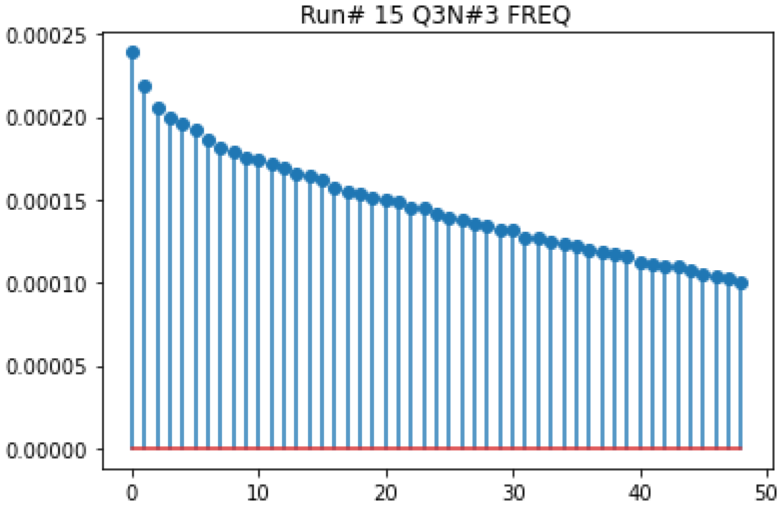

Run# 13 P0N#1 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−134
−136
−138
−140
−142
−144
−146
−148
−150
dB

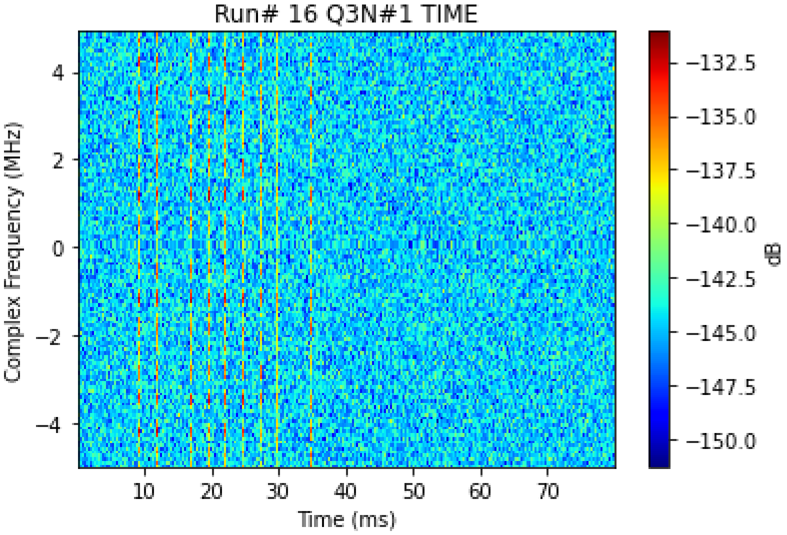

Run# 13 P0N#1 FREQ
0.0008
0.0006
0.0004
0.0002
0.0000
0
10
20
30
40
50

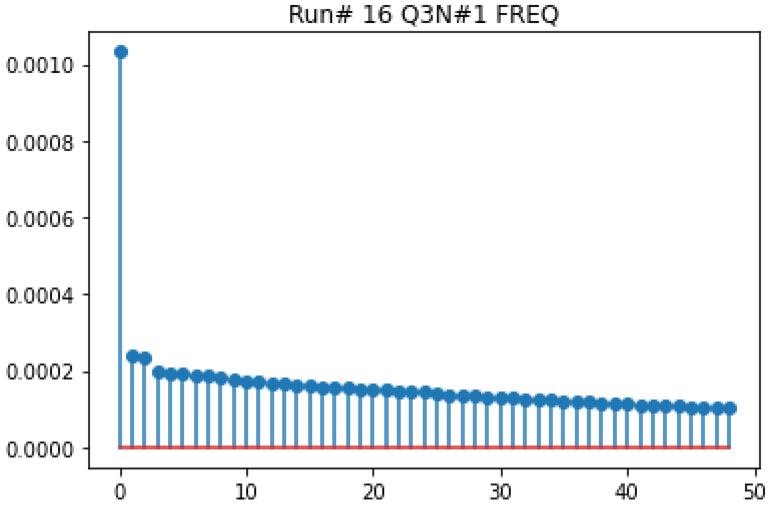

Run# 14 P0N#2 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−125
−130
−135
−140
−145
−150
dB

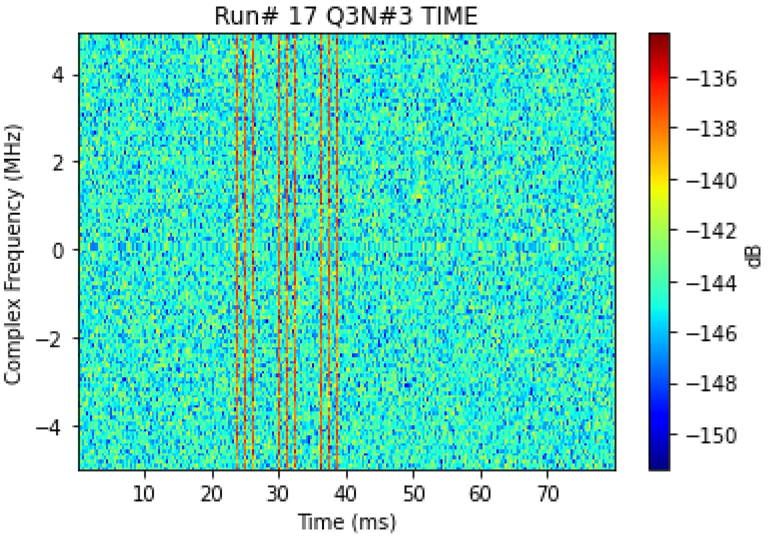

Run# 14 P0N#2 FREQ
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
10
20
30
40
50

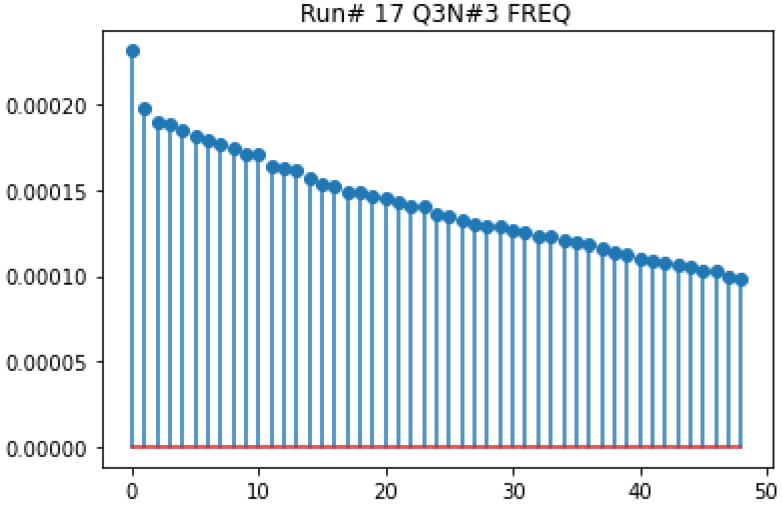

Run# 15 Q3N#3 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−132.5
−135.0
−137.5
−140.0
−142.5
−145.0
−147.5
−150.0
dB

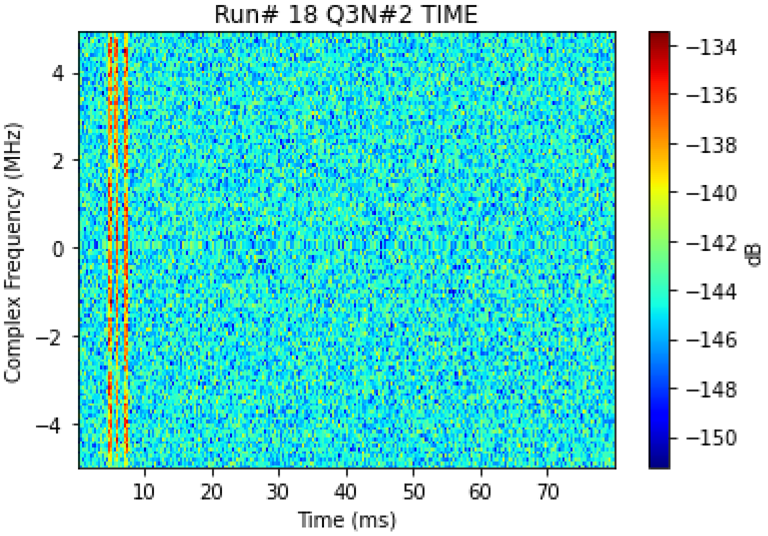

Run# 15 Q3N#3 FREQ
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
0
10
20
30
40
50

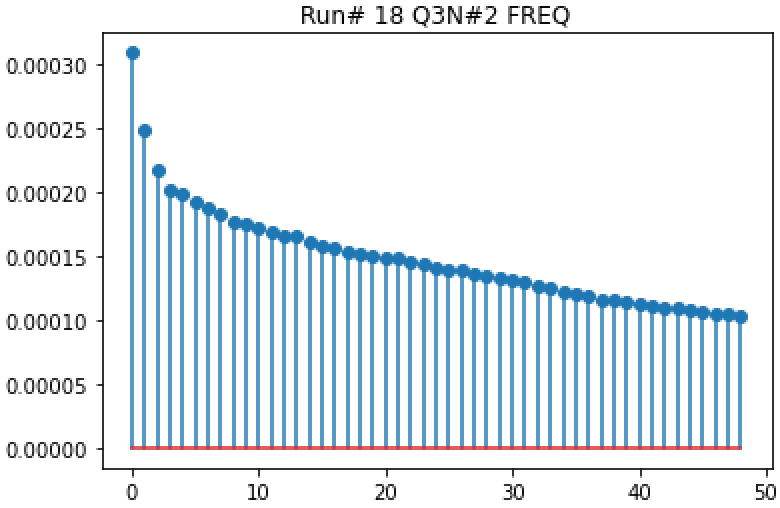

Run# 16 Q3N#1 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−132.5
−135.0
−137.5
−140.0
−142.5
−145.0
−147.5
−150.0
dB

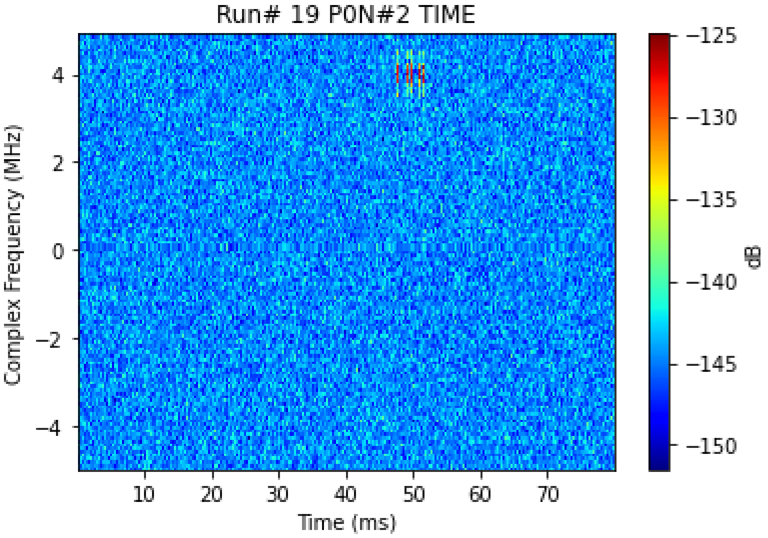

Run# 16 Q3N#1 FREQ
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
10
20
30
40
50

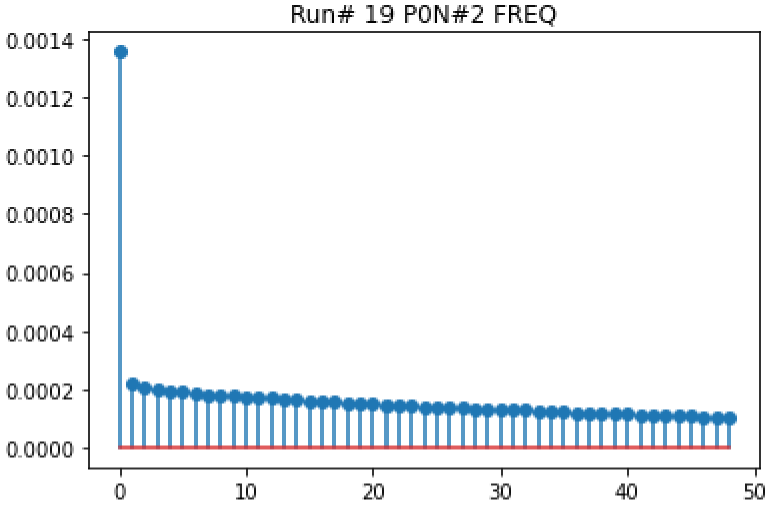

Run# 17 Q3N#3 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−136
−138
−140
−142
−144
−146
−148
−150
dB

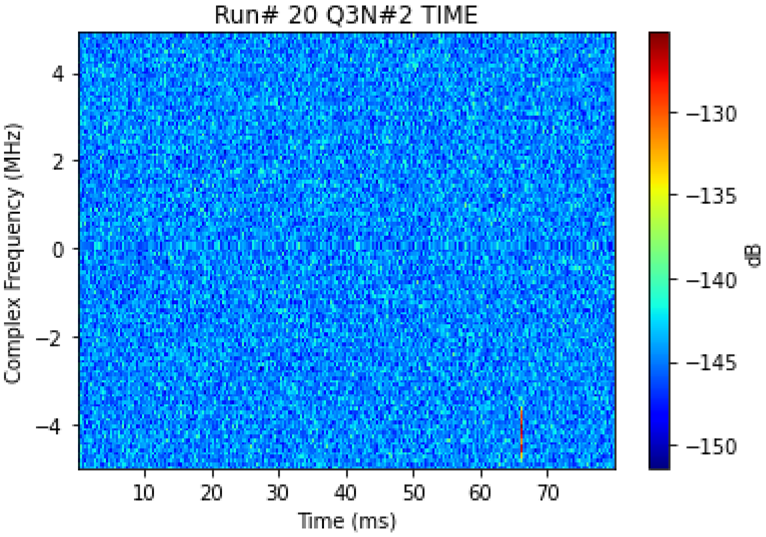

Run# 17 Q3N#3 FREQ
0.00020
0.00015
0.00010
0.00005
0.00000
0
10
20
30
40
50

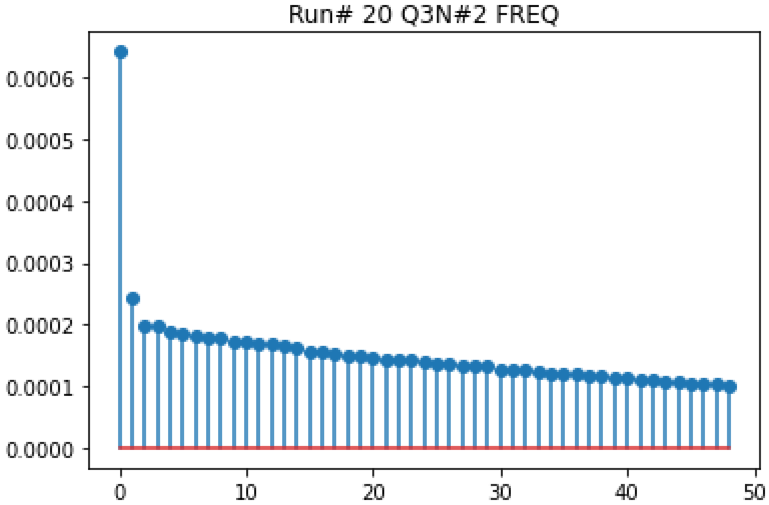

Run# 18 Q3N#2 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−134
−136
−138
−140
−142
−144
−146
−148
−150
dB


Run# 18 Q3N#2 FREQ
0.00030
0.00025
0.00020
0.00015
0.00010
0.00005
0.00000
0
10
20
30
40
50


Run# 19 P0N#2 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−125
−130
−135
−140
−145
−150
dB


Run# 19 P0N#2 FREQ
0.0014
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
10
20
30
40
50


Run# 20 Q3N#2 TIME
Complex Frequency (MHz)
4
2
0
−2
−4
10
20
30
40
50
60
70
Time (ms)
−130
−135
−140
−145
−150
dB


Run# 20 Q3N#2 FREQ
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
10
20
30
40
50